Open Ended Evaluation of Voting Systems

John Kelsey

NIST

March 2006

History and Related Work

- TGDC resolution on OEVT
- Preliminary threats presentation/paper
- OEVT presentation last year
- Lots of informal identification of threats
- NIST Threats to Voting Systems Workshop
- NIST/Brennan Center threat analysis

Overview

- What is open-ended testing?
- Why do we need it?
- How do we plan to do it?
- What do we still need to know?
  - Technical Questions
  - Policy Questions

What is Open-Ended Testing?

- Checklist testing: "Does this system conform to the standard?"
  - Make sure they have right security controls
  - Make sure controls are configured right
- Open-ended testing: "Can I find a weakness in this system?"
  - Look for basic design flaws in system.
  - Look for ways to defeat security controls.

We Need Both

- Checklist testing
  - Like having policeman check out your home security
  - Easy to replicate, “best practices”
  - Need to be competent
- Open-ended testing
  - Like having professional burglar check out your home security
  - Hard to replicate, hard to budget
  - Need to be really good

Examples of Open-Ended Analysis on Voting Systems

- Conceptual:
  - NIST Voting Threats Workshop
  - Brennan Center / NIST Threat Analysis
  - Harris Book
- Specific:
  - Hopkins Report
  - SAIC Report?
  - Various public attacks, e.g., Hursti's attack
  - *RABA Report*

RABA Report: January 2004 (1)

"*A Red Team exercise is designed to simulate the environment of an actual event, using the same equipment and procedures of the system to be evaluated.* Teams are then free to experiment with attack scenarios without penalty. In this fashion a broad range of vulnerabilities may be discovered and validated in an operational environment."

RABA Report: January 2004 (2)

"A Red Team exercise is designed to simulate the environment of an actual event, using the same equipment and procedures of the system to be evaluated. *Teams are then free to experiment with attack scenarios without penalty.* In this fashion a broad range of vulnerabilities may be discovered and validated in an operational environment."

RABA Report: January 2004 (3)

“A Red Team exercise is designed to simulate the environment of an actual event, using the same equipment and procedures of the system to be evaluated. Teams are then free to experiment with attack scenarios without penalty. *In this fashion a broad range of vulnerabilities may be discovered and validated in an operational environment.”*

RABA Report, Cont'd

- Found practical flaws
  - Reuse of system wide password
  - Reuse of system wide keys
  - Poor locks
  - Unpatched GEMS server software
- Once known, all these could be fixed!
  - This is why OEVT is valuable.

Goal: Make Voting Systems Stronger !

- Preparing for test can improve design
  - Documentation requirements!
  - Known vulnerabilities in COTS products
  - Careful design and internal analysis to avoid costly failed tests
- Fix problems that are found
  - Find and fix “low hanging fruit”
  - Address design/procedural flaws

Broad Process:

- Agree on Rules of Engagement
- Submit Documentation
  - If there's a problem, fix it
- Submit Full Sample System
  - Evaluate and report problems to be fixed
- Produce Two Reports
  - Internal report for vendor and EAC
  - External report for public

Rules of Engagement

- What access/resources does attacker have for full attacks?
- What access does attacker have for each intermediate attack goal?
- Are any attacks excluded?
- *Policy Issue: Should this be mostly predefined for everyone, or mostly open for negotiation?*

Submission Package

- Security Documentation
  - Explain how voting system accomplishes security goals specified in standard
- Procedures
  - Explain how normal voting, recounts, and other things are done. Explain why secure.
- *Policy/Technology Issues:*
  - *What procedures are included? How much detail?*
  - *How do we ensure accuracy of submission?*

Full Attacks

- Looking at security documentation, procedures, and full system description, find a way to: (for example)
  - Fix election given some level of insider access.
  - Violate voter privacy w/o cooperation given insider access.
  - Disrupt or discredit election w/ no insider access.
- *Policy/Technology Issue: What should be definition of full attacks? How much should RoE matter?*

Intermediate Attack Goals

- Demonstration of a fundamental flaw in security of voting system
- Examples:
  - Get unauthorized software installed
  - Cause software to run without auth.
  - Cause loggable event w/o entry in event log
  - Get command shell w/o access
- *Technology Issue: What should be intermediate attack goals? How many do we need?*

Why Fail System on Intermediate Attack Goals?

- Intermediate attack goal represents a security requirement in standard.
  - Violate requirement -> not compliant
- Encourage defense in depth
- Save evaluator time/resources
  - Getting full attack takes time
  - Better to use time evaluating rest of system

Pass or Fail?

- Unambiguous Pass
- Unambiguous Fail
  - Incomplete or incorrect documentation
  - Full or intermediate attack goals met
- Grey Areas
  - Standard should minimize ambiguity
- *Policy Issue: Should lab decide pass/fail or should external body do so?*

The Final Reports

- Internal Report to Vendor and EAC
  - If failed, tell vendor broadly how to fix system.
  - If passed, tell vendor what was looked at, and what problem areas were noticed.
- External Report to Public (passed systems)
  - Rules of Engagement, Procedures, Security Documentations
  - List what was looked at, how, and summarize conclusions
  - Include other problem areas noticed

Resource and Money Issues

- *How much is available for open-ended testing?*
  - *What fraction of total testing budget?*
  - *OEVT can be expensive!*
- *Conflict of Interest Issues*
  - *Lab probably paid by vendor, need to minimize conflict of interest*
  - *Lab accreditation, unambiguous standard*

Is This a Feasible Approach?

- Few or no successful examples of this kind of approach!
  - Successful examples are limited in scope
  - Crypto algorithm evaluations
  - Crypto module evaluations
  - Network penetration testing
  - Manual/automatic source code review for security-relevant bugs

Current Plan: Go Slowly

- Some parts of standard can only be tested in open-ended way
  - Security documentation, procedures
- Some open-ended testing relatively straightforward
  - Source code review, vulnerability scans, physical inspection for locks / exposed ports
- Increase resources on OEVT as we gain operational experience running program